Assignment 2 Compiler CSCI 4020U & SOFE 3960U

A. Designing a topdown parser: Consider the following grammar.

prog : stmt-list
stmt-list : stmt-list stmt | stmt
stmt : PRINT expr | PRINT string
string : BEGINQUOTE charlist ENDQUOTE
charlist : charlist LETTER | ε
expr : expr + term | expr - term | term
term : term * factor | term / factor | factor
factor : (expr) | NUM

The nonterminal symbols are:

prog, stmt-list, stmt, string, charlist, expr, term, factor

The terminal symbols are:

PRINT, BEGINQUOTE, ENDQUOTE, LETTER, +, -, *, /, (,), NUM

(A1) The grammar is left-recursive. Transform it into a non-left recursive grammar using the following ranking of the nonterminals.

prog, stmt-list, stmt, string, charlist, expr, term, factor

prog : stmt-list
stmt-list : stmt stmt-list′
stmt-list′ : stmt stmt-list′| ε
stmt : PRINT expr | PRINT string
string : BEGINQUOTE charlist ENDQUOTE
charlist : charlist′
charlist′ : LETTER charlist′| ε
expr : term expr′
expr′ : + term expr′ | - term expr′ | ε
term : factor term′
term′ : * factor term′ | / factor term′ | ε
factor : (expr) | NUM

(A2) Further modify the grammar so it is left factored.

```
prog            : stmt-list
stmt-list       : stmt stmt-list′
stmt-list′      : stmt stmt-list′| ε
stmt            : PRINT word
word            : expr | string
string          : BEGINQUOTE charlist ENDQUOTE
charlist        : charlist′
charlist′       : LETTER charlist′| ε
expr            : term expr′
expr′           : + term expr′ | - term expr′ | ε
term            : factor term′
term′           : * factor term′ | / factor term′ | ε
factor          : ( expr ) | NUM
```

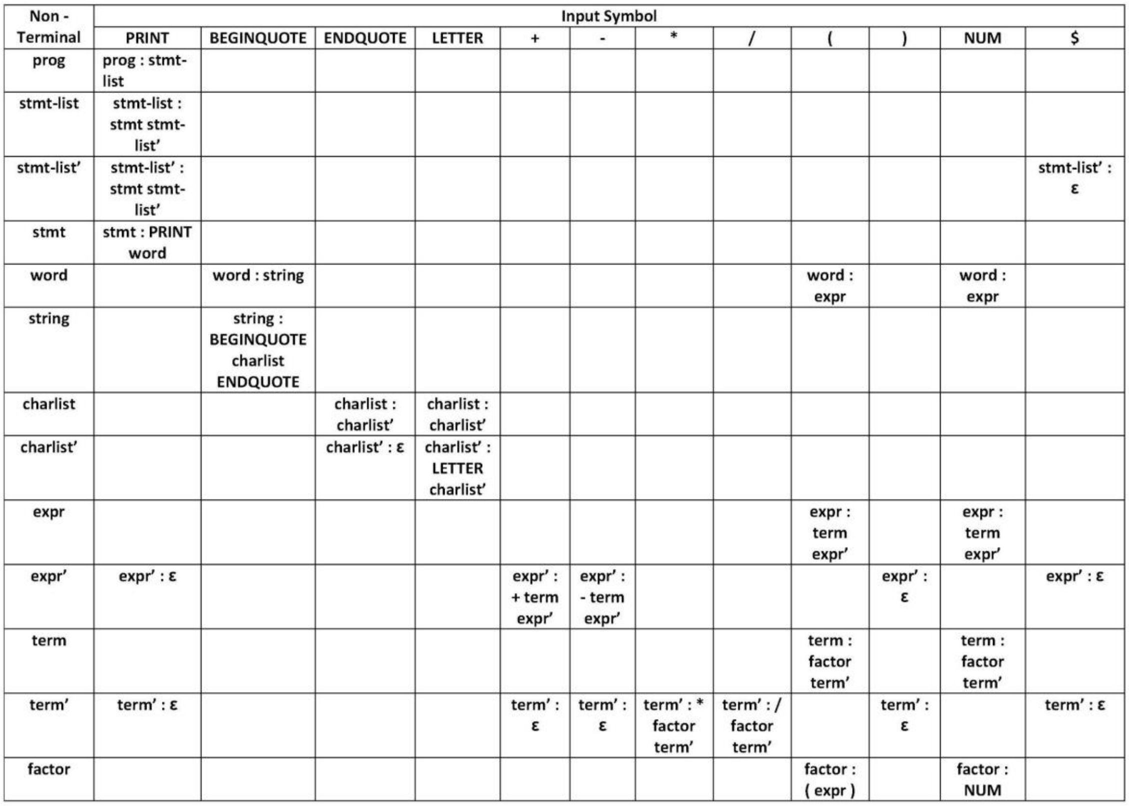
(A3) Compute the FIRST sets of each nonterminal symbols.

prog	**: { PRINT }**
stmt-list	**: { PRINT }**
stmt-list′	**: { PRINT }**
stmt	**: { PRINT }**
word	**: { (, NUM, BEGINQUOTE }**
string	**: { BEGINQUOTE }**
charlist	**: { LETTER }**
charlist′	**: { LETTER }**
expr	**: { (, NUM }**
expr′	**: { +, - }**
term	**: { (, NUM }**
term′	**: { *, / }**
factor	**: { (, NUM }**

(A4) Compute the FOLLOW sets of each nonterminal symbols

prog	**: {}**
stmt-list	**: {}**
stmt-list′	**: {}**
stmt	**: { PRINT }**
word	**: { PRINT }**
string	**: { PRINT }**
charlist	**: { ENDQUOTE }**
charlist′	**: { ENDQUOTE }**
expr	**: { PRINT,) }**
expr′	**: { PRINT,) }**
term	**: { +, -, PRINT,) }**
term′	**: { +, -, PRINT,) }**
factor	**: { *, /, +, -, PRINT,) }**

(A5) Compute the predictive parsing table of the grammar.

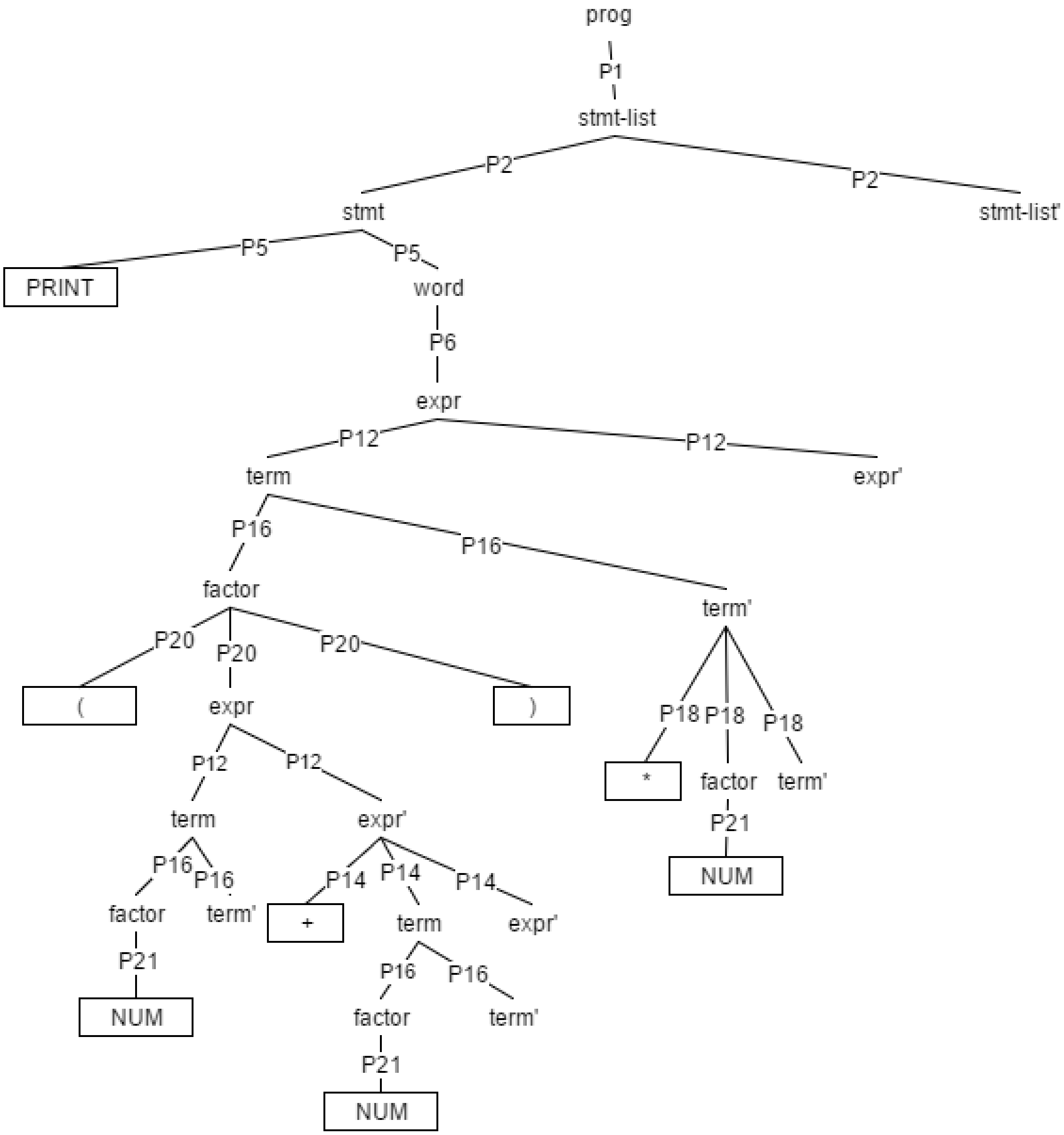


Non - Terminal	Input Symbol											
	PRINT	BEGINQUOTE	ENDQUOTE	LETTER	+	-	*	/	(	)	NUM	$
prog	prog : stmt-list											
stmt-list	stmt-list : stmt stmt-list’											
stmt-list’	stmt-list’ : stmt stmt-list’											stmt-list’ : ε
stmt	stmt : PRINT word											
word		word : string							word : expr		word : expr	
string		string : BEGINQUOTE charlist ENDQUOTE										
charlist			charlist : charlist’	charlist : charlist’								
charlist’			charlist’ : ε	charlist’ : LETTER charlist’								
expr									expr : term expr’		expr : term expr’	
expr’	expr’ : ε				expr’ : + term expr’	expr’ : - term expr’				expr’ : ε		expr’ : ε
term									term : factor term’		term : factor term’	
term’	term’ : ε				term’ : ε	term’ : ε	term’ : * factor term’	term’ : / factor term’		term’ : ε		term’ : ε
factor									factor : (expr)		factor : NUM	

(A6) Consider the following program.

PRINT (NUM + NUM) * NUM

Use the predictive parsing table to construct the parse tree. At each step, show the production used to expand the nodes.



P1. **prog : stmt-list**

P2. **stmt-list : stmt stmt-list’**

P3. **stmt-list’ : stmt stmt-list’**

P4. **stmt-list’ : ε**

P5. **stmt : PRINT word**

P6. **word : expr**

P7. **word : string**

P8. **string : BEGINQUOTE charlist ENDQUOTE**

P9. **charlist : charlist’**

P10. **charlist’ : ε**

P11. **charlist’ : LETTER charlist’**

P12. **expr : term expr’**

P13. **expr’ : ε**

P14. **expr’ : + term expr’**

P15. **expr’ : - term expr’**

P16. **term : factor term’**

P17. **term’ : ε**

P18. **term’ : * factor term’**

P19. **term’ : / factor term’**

P20. **factor : (expr)**

P21. **factor : NUM**